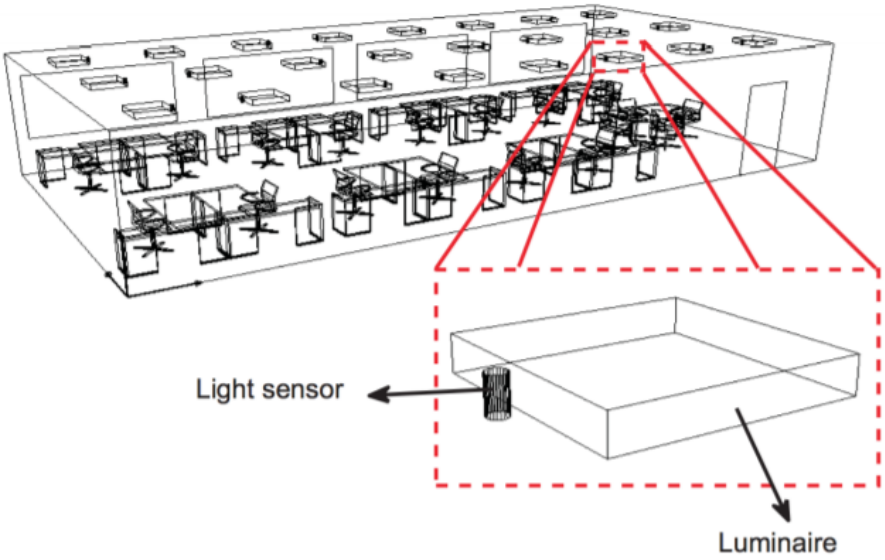
MEEC - Mestrado Integrado em Engenharia Electrotécnica e de Computadores

Distributed Real-Time Control Systems (SCDTR)

2019

Project report

Real-Time Cooperative Decentralized Control of Illumination Systems



Maria Brites, 83842
João Lopes, 83486
Gabriel Antero, 95389

Resumo

Este relatório propõe um modelo de Controlo Distribuído para um sistema de iluminação de secretárias em ambiente fechado, com influência de iluminação externa. Foi criada uma versão do sistema em pequena escala, que conta com três luminárias compostas por 3 LEDs, 3 sensores de iluminação (LDR) e três placas Arduino Uno, que lêem os dados do sensor e controlam a energia fornecida pelo LED com base no valor de iluminação lido. A fim de minimizar a energia consumida pelo sistema, respeitando os limites mínimos de iluminação que devem ser mantidos em cada secretária, foi implementado um algoritmo de optimização que funciona de forma descentralizada. A comunicação entre luminárias é realizada através da rede de controlo CAN-BUS, que permite o envio e recepção de mensagens no sistema. Também foi desenvolvida uma aplicação em $C++$ para que, ao conectar um dos Arduinos ao PC, o utilizador possa receber informações e enviar comandos ao sistema de iluminação. A metodologia e os resultados obtidos são descritos ao longo do relatório, incluindo também destaque em algumas melhorias feitas no controlo, além de comparações entre dois tipos de Controlo Descentralizado: Controlo Cooperativo e Não Cooperativo, verificando-se uma melhor performance do sistema em termos de consumo de energia no segundo caso.

1 Introdução

Nos últimos anos, devido a questões económicas e ambientais, tem sido crescente a busca por sistemas elétricos que minimizem a energia consumida. Em sistemas de iluminação o uso de LEDs (Light Emitting Diodes) permitiu grandes avanços na área uma vez que estes consomem menos energia que as outras tecnologias existentes, nomeadamente as lâmpadas incandescentes e fluorescentes. Devido à sua versatilidade, pequeno tamanho e capacidade de regulação da intensidade os LEDs têm sido cada vez mais utilizados em sistemas de iluminação de cidades, casas, escritórios etc. A sua utilização motivado a criação de "espaços inteligentes" em que a iluminação pode ser controlada por forma a diminuir gastos energéticos e oferecer conforto os utilizadores dos espaços. Uma abordagem a este problema é apresentada em [4].

Neste projecto, será simulado um escritório com três secretárias, em que cada uma possui uma luminária "inteligente". As luminárias são constituídas por um LED, um sensor de luminosidade, e um micro-controlador com capacidade computacional e de comunicação com outras luminárias. O objectivo é usar estas luminárias para criar um sistema de controlo descentralizado cooperativo em tempo real para a iluminação do escritório. O escritório é simulado por uma caixa de sapatos e cada luminária por um Arduino, um LED, e um circuito eléctrico de medição de iluminação.

O objectivo do sistema é minimizar a energia eléctrica consumida e ao mesmo tempo garantir o conforto dos utilizadores do escritório. O conforto dos utilizadores será garantido minimizando o fenómeno de *flicker* que poderá ser induzido devido à existência de iluminação externa ou por interferência causada pelas outras luminárias, bem como controlando a intensidade de cada LED individualmente por forma a que o nível de iluminação, em cada secretária esteja acima de um determinado nível. Num ambiente de escritório, este nível deverá corresponder a 500 lux em secretárias ocupadas e 200 lux em secretárias desocupadas, de acordo com [2]. No entanto, uma vez que no modelo realizado estes valores não são alcançáveis, será definida uma luminância mínima de 50 lux para secretárias desocupadas e de 100 lux para secretárias ocupadas. O consumo de energia será minimizado com recurso a um algoritmo de optimização distribuída, o *Consensus* [3], que será executado utilizando a capacidade de computacional e de comunicação das luminárias.

O relatório seguirá a seguinte estrutura: a definição do problema em estudo é introduzida na secção 2; na secção 3 é apresentada a solução proposta para o problema em questão; na secção 4 são descritos os detalhes da implementação realizada. Segue-se na secção 5 a apresentação e discussão dos resultados experimentais obtidos e finalmente na secção 6 uma conclusão sobre o projecto e sobre a solução implementada é realizada.

2 Definição do Problema

O objectivo deste trabalho é desenvolver um sistema de controlo descentralizado cooperativo em tempo real para a iluminação em escritório que minimize a energia consumida. Este é um problema de controlo óptimo com restrições em que o objectivo é minimizar a energia total consumida por todas a luminárias. As restrições são impostas pelos níveis de referência de iluminação que cada secretária deve ter e pelas restrições físicas do LEDs (não podem gastar energia "negativa"nem gas-

tar mais do que a sua capacidade máxima). Considerando n secretárias, e que na secretária i incide um iluminação l_i, a referência de iluminação é L_i e que a intensidade do respectivo LED é d_i pode-se definir o problema como:

$$\begin{aligned} \min_{d_1 \ldots d_n} \quad & f(d_1, ..., d_n) \text{ tal que} \\ & l_i = \sum_{j=1}^{n} d_i K_{ij} + o_i \geq L_i, \forall i \\ & 0 \leq d_i \leq 100, \forall i \end{aligned} \tag{1}$$

em que o_i é a iluminação externa incidente na secretária i e K_{ij} é a influência da luminária j na iluminação incidente na secretária i. A função de custo $f(d_1, ..., d_n)$ representa a energia total consumida por todas as luminárias no escritório. Considerando c_i o custo da energia na luminária i e $\boldsymbol{c}$ o vector de todos os custos, a função de custo que representa a energia total gasta é

$$f(d_1, ..., d_n) = c_1 d_1 + ... + c_n d_n = \boldsymbol{c}^T \boldsymbol{d}. \tag{2}$$

A unidade de medida desta grandeza é o Joule [J].
Para avaliar a performance do sistema em termos de consumo de energia, será calculada a energia consumida em cada luminária, dada pela acumulação das potências instantâneas ao longo do tempo. Supondo que a potência máxima na luminária j é dada por P_j e que existem K luminárias a função de custo que representa a energia total gasta é

$$E_j = P_j \sum_{i=1}^{N} d_{i-1}(t_i - t_{i-1}) \tag{3}$$

onde i é o indice das amostras de controlo, t_i é o tempo em segundos em que a amostra i foi retirada e d_i é a intensidade relativa da iluminção do LED (é zero quando o LED está desligado e um quando este tem a potência máxima). A unidade de medida desta grandeza é o Joule [J].
Para além de minimizar a energia consumida foi também referido que o conforto do utilizador deve ser garantido. O conforto pode ser descrito por um conjunto de métricas que devem ser controladas, nomeadamente o Erro de de Visibilidade e o Erro de *Flicker*. O Erro de Visibilidade é definido como o erro médio entre a iluminação desejada em cada secretária (L) e a iluminação medida (l) quando a iluminação medida é inferior à desejada. Esta métrica é definida globalmente, para K secretárias, como a soma dos erros médios, V_j, em cada secretária:

$$V = \sum_{j=1}^{K} V_j \tag{4}$$

em que

$$V_j = \frac{1}{N} \sum_{i=1}^{N} max\left(0, L(t_i) - l(t_i)\right) \tag{5}$$

onde N é o número total de amostras usadas para calcular a métrica e t_i o instante temporal de cada amostra em segundos.
O fenómeno de *flicker* é caracterizado por oscilações de grande frequência mas pequena amplitude na iluminação quando a referencia é constante. O *flicker* pode ser definido como

$$f_i = \begin{cases} \left(\|l_i - l_{i-1}\| + \|l_{i-1} - l_{i-2}\|\right)/(2T_s) \\ \qquad \text{se } (l_i - l_{i-1}) \times (l_{i-1} - l_{i-2}) \\ 0 \qquad \text{c.c.} \end{cases} \tag{6}$$

em que T_s é o período de amostragem, l_i é a luminância medida no instante t_i e $|A|$ é o operador que representa o valor absoluto da variável A. Esta fórmula apenas deve ser considerada nos períodos em que a referência se mantém constante, devendo os períodos nos quais existe uma variação explicita do sinal de referência e um consequente regime transitório da resposta ser excluídos. Podemos agora definir o Erro de *Flicker* que é dado em cada secretária por

$$F = \frac{1}{N} \sum_{i=1}^{N} f_i. \tag{7}$$

Mais uma vez esta formula refere-se a cada luminária individualmente. O erro total é dado pela soma da cada um dos erros individuais. As unidades desta métrica são o [LUX/s].

3 Solução Proposta

3.1 Controlo Descentralizado

A solução proposta baseia-se num modelo descentralizado em que nenhum controlador toma decisões globais. Em vez disso cada controlador toma decisões acerca do seu comportamento e o comportamento do

sistema consiste no conjunto dos comportamentos individuais de cada controlador. Cada controlador comunica as suas decisões a todos os outros. No entanto não existe um controlador com conhecimento de todos os parâmetros e que executa todo o processamento do algoritmo. Este tipo de topologia permite uma solução mais rápida em cada iteração do algoritmo de optimização e também dispensa a existência de um nó com grande capacidade de memória, uma vez que os dados estarão repartidos pelos vários micro-controladores. A solução óptima para o problema de optimização, descrito na equação 1, é determinada após um processamento distribuído cujos resultados são comunicados entre os nós e uma solução global é determinada por todos. Para a determinação da solução existem duas abordagens que podem ser seguidas, uma abordagem não cooperativa que permite atingir os objectivos individuais de uma forma não óptima e uma outra cooperativa que permite determinar soluções óptimas.

3.2 Controlo não Cooperativo

Na abordagem distribuída não cooperativa cada nó tenta atingir os seus objectivos individualmente de forma não óptima. As eventuais influências que cada nó possa ter nos restantes nós da rede são interpretadas por estes como perturbações externas ao sistema que tentarão ser eliminadas. Para um sistema de dois nós, conforme o sistema da figura abaixo, esta abordagem funciona da seguinte maneira:

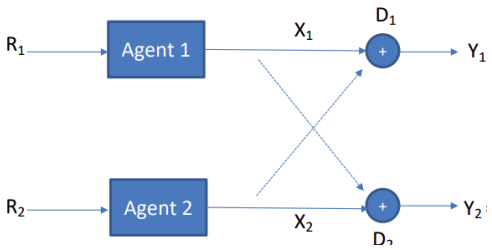


Figura 1: Diagrama de Blocos de um sistema de dois nós.

As variáveis Y_1 e Y_2 podem ser escritas como:

$$\begin{aligned} Y_1 &= D_1 + X_1 \\ Y_2 &= D_2 + X_2 \end{aligned} \tag{8}$$

Ao assumir um acoplamento entre ambos os nós, D_1 e D_2 passam a ser iguais a

$$\begin{aligned} D_1 &= k_{12} * X_2 \\ D_2 &= k_{21} * X_1 \end{aligned} \tag{9}$$

Onde k_{12} e k_{21} são os ganhos relativos ao acoplamento entre os nós.
Portanto, ao analisar como os dois agentes irão reagir, assumindo dois sistemas idênticos realimentados negativamente, percebe-se que o sinal X enviado pelos agentes será sempre uma função da diferença do valor de referência R e de D, que representa o acoplamento com o outro nó.

3.3 Controlo Cooperativo

Tendo em vista a necessidade de que cada agente em um nó realize acções conjuntas para atingir o objectivo comum do sistema descrito na secção anterior, é necessário que os nós realizem comunicações entre si a fim de que encontrem um consenso nos valores das variáveis à cada intervalo de amostragem. Este consenso evita com que o sistema se dirija à uma condição de instabilidade. Sendo assim, é útil utilizar o modo de Controlo Cooperativo entre os nós do sistema. Tal sistema possui como condição primordial uma clara e concisa comunicação entre os nós, para que estes realizem comunicações precisas entre si e que todas as informações possam ser enviadas e interpretadas sem confusões, que podem levar à tomadas de decisões erradas por parte de algum dos nós e que pode gerar alguma instabilidade ou operação errada do sistema.
Além da comunicação, outro detalhe importante a ser levado em consideração é a referência temporal entre os nós. Caso a comunicação entre os nós seja síncrona, estes estarão a realizar cálculos de forma simultânea e espera-se que após cada tempo de amostragem, os nós enviem suas soluções encontradas e outras informações importantes para os outros nós. Caso os cálculos não sejam realizados até que o tempo de amostragem acaba, o sistema não pode ser mais considerado em tempo real e isso levará a erros nos cálculos futuros fazendo com que o sistema não chegue numa solução válida. Para um caso assíncrono, a cada período de amostragem são realizados os cálculos em um nó a parte. Após o fim desse período, o nó envia as informações para o próximo nó, que passa a realizar suas contas com o valor recebido do nó anterior.
Para tal comunicação, será utilizada a rede de controlo CAN-BUS, que permite a transmissão de mensagens a

todos os nós da rede.
Para que o consenso seja realizado, é necessária a implementação de um algoritmo iterativo que leve em consideração o problema geral de optimização, respeitando as restrições globais do problema, como definido na secção anterior.

3.3.1 Algoritmo de Consenso

Para a resolução do problema de optimização descrito, será aplicado o algoritmo *consensus*, um método iterativo que distribui o cálculo pelos nós da rede [3]. Inicialmente decompõe-se o problema a ser optimizado em termos do somatório da função chamada de função Custo que é definida como a multiplicação entre os vectores de custo e os vectores de saída de cada nó.

$$f(\boldsymbol{d}) = \sum_{i=1}^{N} f_i(\boldsymbol{d}) \tag{10}$$

onde a função Custo global é definida como

$$f_i(\boldsymbol{d}) = \boldsymbol{c}_i^T \boldsymbol{d} \tag{11}$$

Além disso, o conjunto de restrições pode ser decomposto como

$$C = \cap_{i=1}^{N} C_i \tag{12}$$

sendo C_i definido por

$$C_i = (\boldsymbol{d} : 0 \leq d_i \leq 100 e \boldsymbol{k}_i^T \boldsymbol{d} \geq L_i - o_i) \tag{13}$$

Portanto, o Algoritmo de Consenso provê uma forma de computar a solução do problema de optimização de forma distribuída ao utilizar um conjunto de nós como elementos individuais em uma rede.Isto é alcançado ao considerar variáveis locais $\boldsymbol{d}_i$ em cada nó i e uma variável global $\overline{\boldsymbol{d}}$ na qual as variáveis locais devem estar de acordo:

$$min \sum_{i=1}^{N} \overline{f_i}(\boldsymbol{d}_i) s.t \boldsymbol{d}_i = \overline{\boldsymbol{d}} \tag{14}$$

Para solucionar problemas de optimização com restrições, é utilizado o algoritmo ADMM (alternated direciton method of multiplier) que forma um Lagrangiano Aumentado. Conforme a equação abaixo:

$$L_i(\boldsymbol{d}_i, \boldsymbol{y}_i) = \overline{f_i}(\boldsymbol{d}_i) + \boldsymbol{y}_i^T(\boldsymbol{d}_i - \overline{\boldsymbol{d}}) + \frac{\rho}{2}\|\boldsymbol{d}_i - \overline{\boldsymbol{d}}\|_2^2 \tag{15}$$

onde $\boldsymbol{y}$ é um vector dos Multiplicadores de Lagrange e ρ é o termo quadrático de penalidade.

A cada iteração são realizados cálculos de minimização em cada nó utilizando os valores da iteração passada. Estes resultados são enviados ao outros nós do sistema, em seguida novos cálculos são feitos com as informações recebidas dos outros nós. Além disso, cada nó i deve armazenar localmente os seguintes atributos:

- O limite inferior da iluminância Local L_i
- A iluminância externa o_i
- O custo local c_i
- Os limites do actuador local (entre 0 e 100)
- Os ganhos de acoplamento de outras luminárias k_i
- O parâmetro de optimização ρ

Durante os cálculos das possíveis soluções, cada nó deve checar se a solução apresentado é possível de ser feita pelo mesmo, ou seja, deve-se testar as soluções encontradas e compará-las com as restrições definidas para as suas condições de operação. Caso as condições não sejam satisfeitas, o algoritmo sinalizar que tal solução não é possível de ser realizada no sistema.As restrições locais podem ser escritas como:

$$C_i : \{\boldsymbol{A}_i \boldsymbol{d}_i \leq \boldsymbol{b}_i\} \tag{16}$$

com

$$\boldsymbol{A}_i = \begin{bmatrix} -\boldsymbol{k}_i^T \\ -\boldsymbol{e}_i^T \\ \boldsymbol{e}_i^T \end{bmatrix} \tag{17}$$

onde

$$\boldsymbol{e}_i^T = \begin{bmatrix} 0 & \cdot 1 \cdot & 0 \end{bmatrix} \tag{18}$$

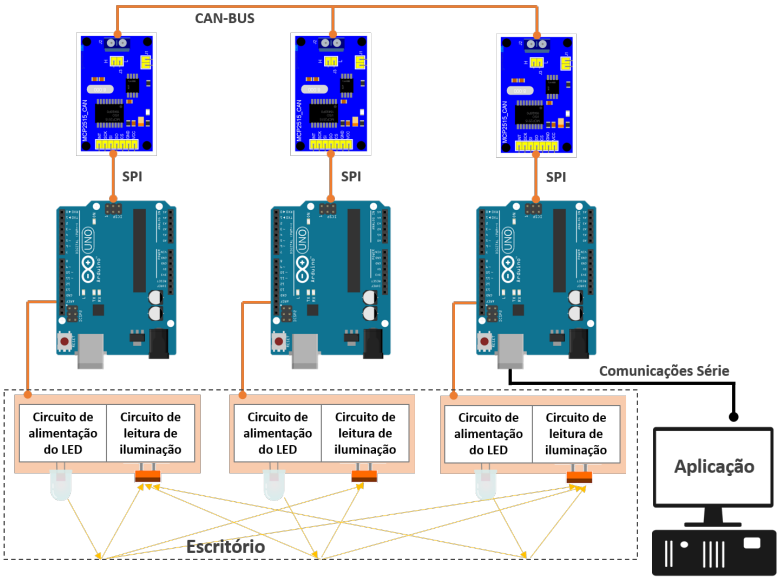
e

$$\boldsymbol{b}_i = \begin{bmatrix} o_i - L_i \\ 0 \\ 100 \end{bmatrix} \tag{19}$$

Sendo assim, o processo de verificação corresponde em confirmar se a solução possível $\boldsymbol{d}_i$ comprova a seguinte desigualdade:

$$\boldsymbol{A}_i \boldsymbol{d}_i \leq \boldsymbol{b}_i \tag{20}$$

Portanto, o Algoritmo de Consenso realiza diversas iterações em que em cada uma é realizado o cálculo óptimo de d_i. Como a função custo é uma função quadrática, as soluções devem ser localizadas no interior do conjunto, sendo assim uma solução global, ou, na fronteira. Logo, d_i é calculado baseado nas restrições impostas pelas fronteiras da função custo. Após



(a) Diagrama global do sistema

(b) Interior e exterior da caixa usada para modelar o sistema

Figura 2: Sistema usado para modelar o escritório com iluminação "inteligente" em pequena escala

o cálculo de d_i é verificado se a solução encontrada é factível, caso esta seja, o valor da solução é salvo. O custo também é calculado e, caso seja menor que o valor pré definido de custo óptimo, este é actualizado e passa a ser o novo custo óptimo.

4 Desenvolvimento

4.1 Arquitectura do Sistema

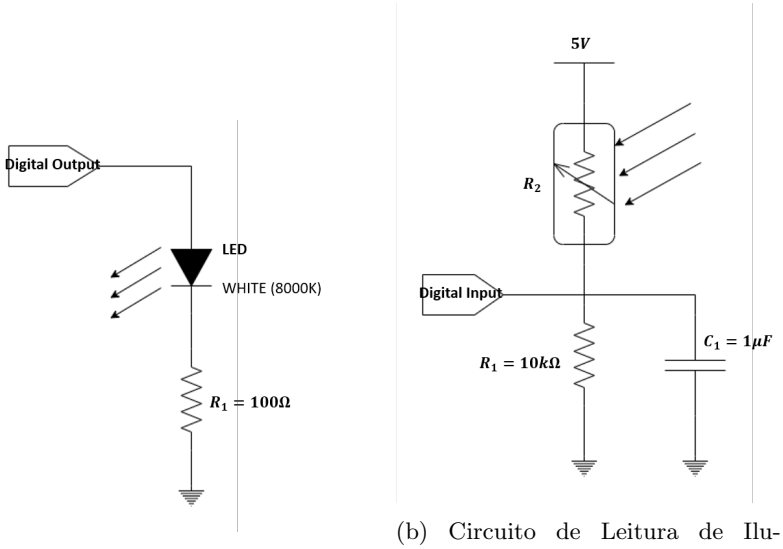
O escritório foi simulado usando uma caixa de sapatos contendo 3 luminárias como mostra a figura 2. Para construir o sistema foram utilizados os seguintes componentes:

- 3 Arduino Uno
- 3 Light Emitting Diode (LED YSL-R547W2C-A13)
- 3 Light Dependent Resistor (LDR GL5528)
- 3 resistências de 100Ω
- 3 resistências de $100k\Omega$
- 3 condensadores de $1\mu F$
- 3 módulos MCP 2515 Can-bus

As componentes do sistema encontram-se descrita nas seguintes secções.

4.2 Modelo da Luminária

Cada luminária é composta por um Ardunino Uno, um circuito de alimentação de um LED, e um circuito para medir a iluminação com um LDR. Cada luminária mede a luminância na respectiva secretária e usa o controlador implementado na placa Arduino para controlar a intensidade do respectivo LED. A figura 3 apresenta os circuitos presentes em cada luminária para alimentação do LED e para leitura da luminosidade.
A intensidade de cada LED é definida através do sinal PWM (Pulse-Width Modulation) que lhe é aplicado. O PWM é a forma que o Arduino utiliza para simular uma fonte de alimentação constante criando um sinal de alta frequência com amplitude fixa mas com *duty-cycle* variável. Ao variar o *duty-cycle* é possível variar a quantidade de energia transportada na onda simulando assim mudanças de amplitude numa fonte de alimentação constante. Uma vez que é usada uma onda de alta frequência é introduzido algum ruído no circuito. Para colmatar este efeito é introduzido um condensador no circuito que funciona como um filtro passa-baixo, reduzindo o ruído na saída do circuito.



(a) Circuito de alimentação do LED

(b) Circuito de Leitura de Iluminância

Figura 3: Circuito do LED e circuito do LDR

4.2.1 Circuito de Leitura

O circuito de leitura de luminosidade é responsável por converter uma tensão medida aos terminais do sensor LDR para as unidades SI da iluminação [LUX]. A resistência do LDR varia consoante a iluminação à qual se encontra exposta, diminuindo com o aumento da iluminação. Isto pode ser observado através da seguinte expressão da tensão de saída do circuito da figura 3 em regime estacionário.

$$V = V_{cc}\frac{R_1}{R_1 + R_2(x)} \Rightarrow R_2(x) = R_1\left(\frac{V_{cc}}{V} - 1\right) \quad (21)$$

onde x é a iluminação incidente no LDR (em LUX), V_{CC} é a tensão de alimentação (5V) e V é a tensão medida no pin analógico. Tendo em conta a característica de "Iluminação *vs* Resistência" presente no datasheet do LDR [5], obtém-se a seguinte relação entre a iluminação e a resistência R_2.

$$log(R_2(x)) = mlog(x) + b. \quad (22)$$

Os parâmetros m e b foram inicialmente estimados através da datasheet. No entanto esta estimativa inicial foi melhorada realizando o processo de calibração. Este processo requer a determinação do valor do parâmetro m para o qual relação entre o *duty-cycle* aplicado ao LED que ilumina o LDR e a iluminação medida por este é linear. Isto é, em regime estacionário, incrementos na actuação do LED que ilumina o LDR devem corresponder a incrementos idênticos no valor de luminância medida. Desta forma foram aplicados valores crescentes de *duty-cycle* em cada um do LED's e foi registada o valor da reticencia LDR no respectivo circuito de leitura. Usando a expressão 22 ajustou-se o valor de m até se obter uma relação o mais linear possível como demonstrado na figura 4.

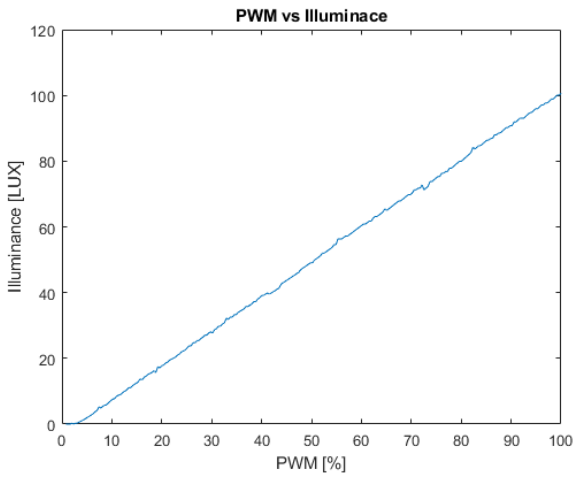


Figura 4: Relação linear entre actuação no LED e luminância medida

O parâmetro b, por sua vez, foi ajustado com recurso a outros luxímetros já calibrados por forma a que nas mesmas condições de iluminação estes medissem o mesmo valor. Para os três circuitos de leitura usados os seguintes parâmetros m e b foram determinados:

- $m_1 = -0.85$
- $b_1 = 6.15$
- $m_2 = -0.85$
- $b_2 = 5.85$
- $m_3 = -0.58$
- $b_3 = 4.51$

4.2.2 Sistema de Primeira Ordem Equivalente

O sistema de iluminação pode ser descrito por um sistema dinâmico de primeira ordem. Nesta formulação a entrada do sistema é a actuação do LED e a saída é a medida do sensor LDR. A relação entrada/saída pode então ser expressa com recurso à transformada de Laplace como uma função de transferência:

$$G(s) = \frac{K_o(x)}{1 + s\tau(x)} \tag{23}$$

onde x é a iluminação desejada, $K_o(x)$ é o ganho estático e $\tau(x)$ é a constante de tempo do sistema. Como se pode ver a função de transferência do sistema depende do valor final da luminância incidente no LDR. A expressão da constante de tempo pode ser derivada analisando o circuito da figura 3. Considerando o circuito um sistema dinâmico é possível estabelecer a seguinte relação

$$\dot{v}\tau(x) = v + V_{CC}\frac{R_1}{R_1 + R_2(x)} \tag{24}$$

com,

$$\tau(x) = R_{eq}(x)C_1 \quad \text{e} \quad R_{eq}(x) = \frac{R_1 R_2(x)}{R_1 + R_2(x)} \tag{25}$$

Para um dado valor de iluminação pretendida x, a tensão v na saída do circuito corresponde à resposta ao escalão de um sistema de primeira ordem, pelo que a solução da equação 24 é dada por

$$v(t) = v_f - (v_f - v_i)e^{-\frac{t-t_i}{\tau(x)}} \tag{26}$$

em que t_i é o instante de tempo da alteração; v_i o valor inicial de v; v_f o valor final de v (estado estacionário) e x_f o valor final de x (estado estacionário).
Este modelo é fortemente dependente do valor final de iluminancia x. Para ilustrar tal dependência efectuaram-se medições do valor da iluminação para diferentes valores impostos de PWM. Na figura 5 encontram-se os resultados.

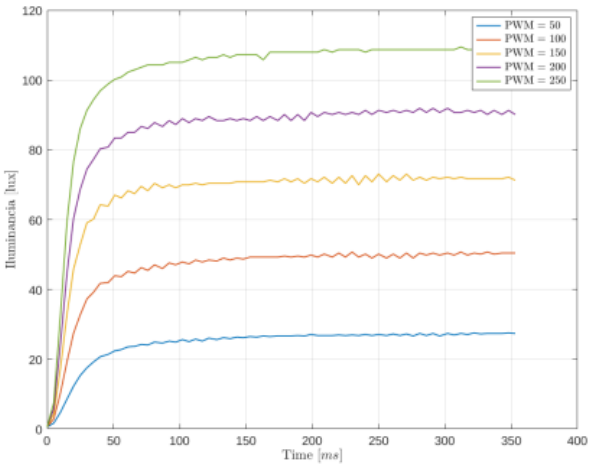


Figura 5: Resposta do sistema para diferentes valores de PWM

A estimação do valor do ganho estático $K_o(x)$ é feita através do quociente entre o valor do sinal de saída após o regime transitório e o valor do sinal de entrada. A constante de tempo $\tau(x)$, por sua vez, é estimada como o tempo que a resposta do sistema demora a atingir aproximadamente 63% do seu valor final.
Para determinar estes parâmetros, e uma vez que eles dependem do valor final da luminância que incide sobre o LDR, foi realizado um "varrimento" para vários valores de PWM. Isto é, realizaram-se experiências em que sucessivamente foram aplicados diferentes intensidades no LED e mediram-se os correspondentes valores de τ, sendo que após a medida o LED é desligado e de seguida novamente ligado com uma intensidade diferente. Na figura 6 apresentam-se os resultados deste processo onde é possível ver a relação entre a constante de tempo e o valor final de luminância. Esta relação pode ser aproximada por um polinómio de segundo grau a partir dos dados apresentados na figura 6. Esse polinómio é $\tau(x) = 0,001x^2 - 0,3834x + 54,962$ para a luminária 1.

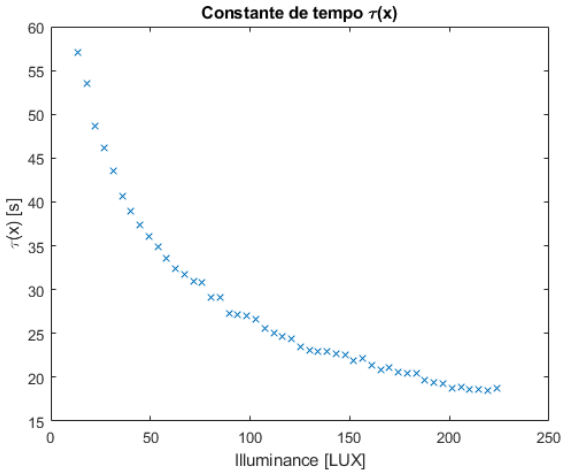


Figura 6: Constante de tempo em função da iluminância

4.3 Controlador Local

No funcionamento normal do sistema, a cada luminária é atribuída uma referência na forma de intensidade luminosa a seguir. O seguimento da referência é garantido através de um controlador, que é executado a uma frequência de $100Hz$. Este controlador é constituído por dois componentes fundamentais: controlador Feedforward e controlador Feedback. A figura 7 apresenta um diagrama de blocos detalhado do controlador implementado.

4.3.1 Controlador FeedForward

O objetivo do controlador Feedforward é controlar a intensidade luminosa do LED em malha aberta afim de alcançar um primeiro valor de iluminação de referência. Naturalmente, uma vez que se trata de controlo sem realimentação, este valor não será exactamente o pretendido devido a perturbações externas e a eventuais erros de modelação. No entanto a sua presença na arquitectura do sistema é de grande importância pois permite acelerar a resposta do sistema. O valor de saída do controlador Feedforward, f_f, é obtido através do modelo do sistema e corresponde ao valor de PWM correspondente à iluminação de referência a seguir. Esta relação é expressa por

$$f_f = \frac{x - o}{G_o} \tag{27}$$

onde x é a iluminação pretendida, o é iluminação exterior e G_o é o ganho que traduz a relação entre a % de sinal PWM aplicado no LED e a luminância medida no circuito LDR.

4.3.2 Controlador FeedBack

O termo FeedBack implementado corresponde a um controlador Proporcional Integral (PI). O objectivo deste controlador é corrigir o valor do termo de Feedforward de forma a levar o sistema a seguir a referência desejada. Este controlador consiste num termo proporcional e outro integral, dado por:

$$u(t) = \underbrace{K_p e(t)}_{p(t)} + \underbrace{K_i \int_0^T e(t)dt}_{i(t)} \tag{28}$$

em que $e(t)$ é o erro (diferença entre a saída do simulador e a tensão medida). Uma vez que o controlador implementado funciona em tempo discreto, usa-se uma transformação bilinear para implementar termo FeedBack descrito na equação 28. O termo proporcional produz o valor que é proporcional à amplitude do erro $e(t)$ aumentando assim a velocidade de resposta. Tem como contrapartida, a possibilidade de criação de "overshoot"e aumentar a frequência de oscilação, ou mesmo tornar o sistema instável.
O termo integral, por sua vez, é proporcional ao erro acumulado. É essencial para a eliminação do erro em regime estacionário gerado pelo termo proporcional e acelera a resposta do sistema, permitindo-o chegar ao valor de referência mais rapidamente.

4.3.3 *Anti-Windup*

O fenómeno de *windup* deriva do facto de todos os actuadores terem limites. No caso do sistema em estudo os limites do LED são a sua intensidade máxima de luminosidade e a sua intensidade mínima, que corresponde a estar desligado. O facto de se usar um termo integral juntamente com um actuador com limites pode originar situações em que o actuador se encontra "saturado"e o integrador continua a fazer a sua função de integrar o erro. No sistema em questão um exemplo desta situação é, por exemplo, quando a intensidade luminosa colocada como referência no controlar está para lá da capacidade luminosa do LED. Nessa situação é impossível que a referência seja alcançada e por isso o erro continuará a ser infinitamente acumulado. Naturalmente que quando a referência mudar para um valor alcançável o "erro acumulado"no integrador será

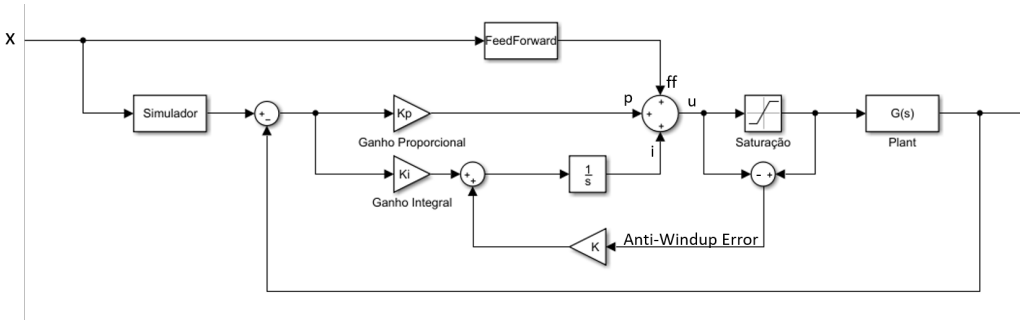


Figura 7: Diagrama de Blocos do Controlador

tão grande que este não terá capacidade para reagir à mudança de referencia.
Para contornar este problema, coloca-se um sistema de *Anti-Windup* cuja função é saturar o termo integral do controlador, evitando assim que o erro acumulado cresça ininterruptamente. O sistema de *Anti-Windup*, quando o actuador se encontra saturado, re-calcula o termo integral do controlador de forma que este novo valor esteja dentro dos limites de saturação.

4.3.4 *Deadzone* e Filtro Passa-Baixo

Uma vez que o conversor analógico-digital do arduino usa 10bits e tem um alcance de 5V, a máxima resolução alcançável é aproximadamente 5mV. Ora pequenas oscilações em torno de um valor de tensão constante podem no processo de digitalização ser ampliadas criando oscilações na resposta do sistema, isto é, o efeito de *flicker*, explicado na secção 2. Para minimizar este efeito implementou-se uma *deadzone* no sinal de erro que alimenta o controlador e um Filtro Passa-Baixo nas leituras de tensão.
A implementação do filtro Passa-Baixo consiste em recolher 5 amostras de cada vez que se pretende ler uma tensão no "pin"analógico do arduino. Após recolhidas as amostras realiza-se a média e este valor corresponderá ao valor da leitura. Foi escolhido recolher 5 amostras para que o tempo de leitura não fosse exagerado ao ponto de comprometer a frequência de trabalho do controlador. A *deadzone* implementada permite que pequenos valores de erro (inferiores a 1 LUX) sejam ignorados. A vantagem de usar um sistema deste tipo é a redução do efeito do ruído e dos erros de quantitização em regime estacionário. Em contrapartida este sistema pode levar ao aumento do erro estático, uma vez que pequenas diferenças entre o sinal de saída e a referência serão ignoradas. Esta é, no entanto, uma desvantagem que se aceita por forma a minimizar o fenómeno de *flicker*

4.3.5 Simulador

Se a referencia do controlador FeedBack for o valor final de iluminação, o sistema pode sobre-reagir a mudanças bruscas na referencia. Isto acontece uma vez que o sistema tem naturalmente um atraso na resposta e ambos os as componentes FeedForward e FeedBack do controlador estão ativos e a funcionar de forma desacolpolada. Assim quando o controlador FeedForward impõe uma mudança "brusca"na entrada do sistema (intensidade do LED) esta será detectada pelo sensor com algum atraso criando assim um "falso"erro de seguimento que o controlador de FeedBack tentará corrigir.
Por forma a contornar esta situação implementa-se um simulador com o objectivo de simular da resposta ao escalão do sistema. O valor de entrada desde simulador será a referencia de FeedForward e a saída será usada para comparar com o valor actual da saída do sistema (neste caso a luminosidade na caixa/escritório) e assim calcular o erro de seguimento usado pelo controlador de FeedForward. Como já foi explicado na secção 4.2.2 o sistema em estudo é um sistema de 1ª ordem pelo que as respostas simuladas são da forma mostrada na equação 26.
A implementação do simulador permite evitar sobre-respostas quando existem mudanças bruscas na referencia de iluminação porque os controladores deixam de funcionar de forma desacoplada.

4.4 Controlador distribuído

Após a integração das três luminárias no sistema, foi implementado o sistema de controlo distribuído coo-

perativo, com recurso ao algoritmo *consensus* descrito na secção 3.3.1. Para que todos os nós tenham acesso ao valor da variável global $\overline{\boldsymbol{d}}$ (que corresponde à média das soluções calculadas por cada nó) é necessário que a cada iteração do algoritmo cada nó envie as soluções $\boldsymbol{d}_i$ por ele calculadas e receba as soluções calculadas pelos outros nós. Para realizar tais tarefas, uma máquina de estados, ilustrada na figura 8 foi desenvolvida no código implementado nos Arduinos.

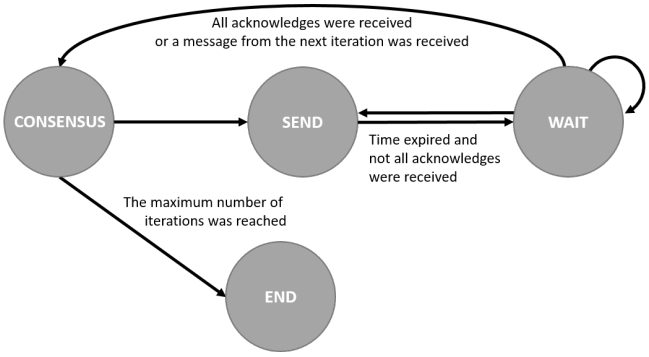


Figura 8: Máquina de estados construída para implementação distribuída do algoritmo Consensus.

Cada vez que é necessário o cálculo de uma nova solução, o nó começa por calcular as suas soluções (estado Consensus). Em seguida, envia-as para cada um dos outros nós (estado Send) e espera por uma confirmação da sua recepção (estado Wait). Caso o tempo de espera seja esgotado e não tenham sido recebidas todas as confirmações nem nenhuma mensagem referente à iteração seguinte, os resultados voltam a ser enviados (estado Send). Caso tenham sido recebidas todas as confirmações ou uma mensagem da iteração seguinte (significando que um dos outros nós já recebeu todas as confirmações da iteração corrente e calculou os resultados da próxima), o nó volta ao estado Consensus e calcula os resultados da próxima iteração.
O algoritmo termina quando é atingido o número limite de iterações, que foi definido como 50. Quando isto ocorre e um dos nós chega ao estado End, este envia aos restantes uma mensagem se sinaliza o fim do *consensus*.
A solução do consensus permite obter o novo valor a utilizar no feedforward, assim como o novo valor de referência a utilizar no sistema de controlo, que é $L_i = \sum_{j=1}^{N} K_{ij} d_{ij} + o_i$, em que K_{ij} é o ganho de acoplamento da luminária i com a luminária j.
A necessidade de implementação de uma máquina de estados resulta do facto de ser necessário continuar a executar o código do controlador individual a cada 10ms enquanto são determinadas as soluções do algoritmo. Uma vez que não é possível fazer todas as computações e envios de mensagens necessários antes do início de um novo ciclo de controlo quando há uma mudança de referência, é necessário guardar o estado de execução atual, de modo a que o algoritmo após o ciclo de controlo seguinte.

4.5 Inicialização e Calibração

Cada vez que o sistema é ligado é executada uma rotina de calibração com o objectivo de determinar os ganhos de acoplamento entre cada uma das luminárias. Estes ganhos expressam a relação entre a intensidade aplicada nos LEDs com o valor de intensidade luminosa medida no sensores LDR. Considerando que o sistema é composto por N luminárias existem N^2 ganhos de acoplamento a ser determinados durante a sequência de calibração. Para além destes ganhos a rotina de calibração permite ainda medir a iluminação externa que afecta cada uma das luminárias.
O efeito de acoplamento entre as luminárias e a influencia da iluminação externa encontra-se expressa na seguinte equação:

$$\begin{bmatrix} L_1 \\ L_2 \\ \vdots \\ L_N \end{bmatrix} = \begin{bmatrix} K_{11} & K_{12} & \dots & K_{1N} \\ K_{21} & K_{22} & \dots & K_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ K_{N1} & K_{N2} & \dots & K_{NN} \end{bmatrix} \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_N \end{bmatrix} + \begin{bmatrix} o_1 \\ o_2 \\ \vdots \\ o_N \end{bmatrix}. \tag{29}$$

Note-se que cada nó apenas determinará e guardará uma linha da matriz de ganhos e a respectiva iluminação externa, pois apenas estes ganhos são necessários para o algoritmo de controlo distribuído explicado na secção 4.4 .Isto é cada nó apenas necessita de saber a influencia que as outras luminárias tem nele não precisando de saber a influencia que este terá nas outras luminárias.
A rotina de calibração começa com a leitura da memória EEPROM dos Arduinos. Nesta memória são guardados previamente os parâmetros m e b, apresentados na secção 4.2.1, característicos de cada luminária. Existe ainda em cada Arduino um identificador guardado que identifica inequivocamente cada luminária no sistema. De seguida é realizado um período de 4 segundos em que todas as luminárias enviam repetitivamente uma mensagem para o CAN-BUS com o seu identificador. Após este período cada uma das luminárias conhecerá todas as outras lumininárias existentes na rede.

Cada luminária mede em seguida a iluminação que nela incide com todas as luminárias desligadas. Esta luminosidade corresponde à iluminação externa. De seguida, segue-se uma rotina em que cada luminária é ligada de cada vez de forma sequencial (seguindo a ordem dos seus identificadores) com a intensidade máxima durante um segundo. Quando uma luminária está acesa, todas as outras medem a intensidade luminosa que nelas incide e assim calculam a influencia que a luminária acesa tem nelas.
Para além do tempo de leitura da memória EEPROM (que é desprezável face ao restante) a sequência de calibração demora $2N + 4$ segundos a executar sendo N o numero de secretárias no sistema. De cada vez que uma nova secretária entra ou sai da rede a sequência de calibração é executada novamente.

4.6 Comunicações

A integração de várias luminárias no sistema requer a definição de um protocolo de comunicações entre os vários nós. Para o efeito foi utilizado o módulo CAN-Bus MCP 2515 e a Serial Peripheral Interface (SPI) do Arduino. A SPI é um protocolo de dados síncrono que permite a comunicação rápida com dispositivos periféricos a curtas distâncias. O módulo MCP2515 implementa o protocolo CAN V2.0B a 1MB/s, possuindo 2 buffers de recepção e 3 buffers de transmissão. A conecção entre cada módulo MCP 2515 e cada Arduino é descrita na figura 2. A interface com o módulo MCP2515 foi realizada utilizando a biblioteca *Arduino MCP2515* [1]. Foi definida uma baudrate de 1000KBPS para as comunicações na rede, a velocidade máxima suportada.
A estrutura das mensagens standard do protocolo CAN é apresentada na figura 9a. Cada mensagem corresponde a $44 + 8N$ bits de informação, em que N é o número de bytes usados do *Data Field*. Caso seja usada a capacidade máxima, a mensagem é enviada em:

$$t = \frac{44 + 8 * 8}{1000000} = 0.108ms \tag{30}$$

Na implementação realizada apenas são enviadas mensagens com um máximo de 32 bits, logo o tempo de envio reduz se a:

$$t = \frac{44 + 8 * 4}{1000000} = 0.076ms \tag{31}$$

A estrutura do identificador na implementação realizada foi definida como mostra a figura 9b. Em particular, os 2 bits menos significativos são usados para identificar o nó que envia a mensagem, e os 2 bits seguintes para identificar o nó ao qual a mensagem se destina (caso estes bits contenham o valor 0, a mensagem é destinada a todos os nós). Quando uma mensagem sem erros é detetada no CAN bus, o seu identificador é comparado com um conjunto de filtros definidos pelo utilizador e a mensagem é movida para um dos buffers de recepção caso verifiquem as condições impostas pelos filtros.
No caso deste sistema, a recepção de mensagens é um ponto crítico, uma vez que facilmente os buffers podem ficar cheios, levando a que o nó perca mensagens enviadas. De modo a impedir que os buffers fiquem sobrecarregados com mensagens, foi definido um filtro específico para cada nó, que aceita apenas identificadores cujo ID do receptor da mensagem seja igual ao ID do nó. Foi ainda definido um filtro em todos os nós que permite aceitar mensagens com o valor 0 nos mesmos bits do identificador. Os sete bits restantes são usados para codificar o tipo de mensagem. Em particular, foram definidos os seguintes tipos de identificadores:

- 0x00* - mensagens da sequência de inicialização e calibração
- 0x01* - mensagens com resultados do *Consensus*
- 0x02* - mensagens de acknowledge do *Consensus*
- 0x03* - fim do *Consensus*
- Os restantes identificadores até 0x7F* codificam pedidos da aplicação do PC

(a) Estrutura das mensagens CAN standard

Identifier – 11 bits		
7 bits	2 bits	2 bits
Message ID	Receiver ID	Sender ID

(b) Estrutura do identificador na implementação realizada

Figura 9: Mensagens CAN

Note-se que esta implementação facilmente poderia ser extendida a mais luminárias com a utilização da Extended Frame do protocolo CAN, que possui identificadores com 29 bits. Assim, estariam disponíveis 11 bits para identificar o nó que envia a mensagem e 11 bits

para identificar o nó ao qual a mensagem se destina, permitindo acomodar uma rede com 120 luminárias.
Para obter as soluções do algoritmo de optimização na implementação realizada, são necessárias 50 iterações. Uma vez que em cada iteração, cada nó tem de enviar pelo menos 2 mensagens, é necessário dispender no mínimo $0.076 \times 50 \times 2 = 7.6ms$ em comunicações. Isto reforça a importância da implementação da máquina de estados descrita na secção 4.4, uma vez que com os tempos de computações adicionais que são necessários é ultrapassado o tempo de amostragem do controlador.

4.7 Aplicação

Foi desenvolvida um interface computacional na linguagem de programação $C++$ para permitir uma interação do sistema com o exterior. A aplicação inclui uma consola que permite que um utilizador interaja com o sistema através do teclado do computador. O computador funciona assim como servidor neste sistema, tendo como função fornecer a informação pedida pelo cliente (o utilizador) bem como transmitir à rede ações a executar.
A comunicação entre a rede de luminárias e o computador é efetuada pela biblioteca *Serial* do Arduino. Apenas um dos nós se encontra ligado ao computador. No entanto, qualquer nó pode ter essa função. Caso sejam detetadas comunicações Serial, o Arduino passa a executar uma "hub function", que processa a mensagem recebida e verifica se esta se dirige a si ou a outro dos nós da rede. No primeiro caso, devolve imediatamente a resposta via comunicação *Serial*. No segundo, envia uma mensagem para o CAN-Bus e quando recebe a resposta referente a essa mensagem devolve-a via comunicação *Serial*.
A aplicação foi desenvolvida usando os serviços I/O assíncronos da biblioteca Boost ASIO, nomeadamente as suas classes leituras e escritas assíncronas. Um servidor assíncrono tem várias vantagens comparativamente a servidores síncronos. Uma das mais importantes é o facto de o servidor não bloquear quando recebe uma mensagem. Ao receber os pedidos é criada uma "fila de espera"para processar as mensagens a receber e enviar. As tarefas de escrita e leitura são processadas assincronamente através de um mecanismo auxiliar fazendo assim com que o processo onde corre o servidor não bloqueie. Assim que as tarefas em "fila de espera"são realizadas é lançado um aviso que notifica o servidor que a respectiva tarefa foi completa. Para o funcionamento da aplicação foram utilizadas classes internas da própria biblioteca Boost ASIO, como as seguintes classes:

- io_context;
- serial_port;
- deadline_timer;
- streambuf;
- const_buffer.

As classes utilizadas da biblioteca Boost ASIO foram fundamentais para o funcionamento da aplicação pois elas permitem o uso da comunicação serial, além de estabelecer um buffer para leitura e escrita entre o servidor e os sistema.

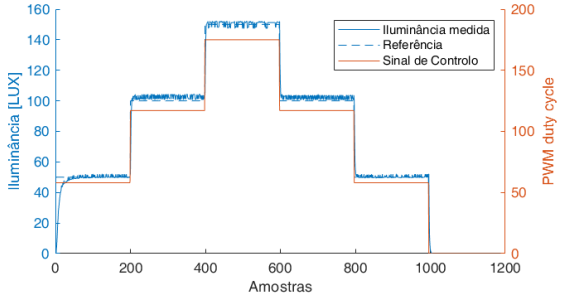
Portanto, pode-se resumir o funcionamento da aplicação ao destacar os seguintes pontos:

- Comunicação: A comunicação entre o computador e o sistema é feita através de conexão serial, como já dito anteriormente. Além disso, foi utilizada a biblioteca Boos ASIO para utilizar seus serviços de comunicação assíncrona.
- Classes: As classes utilizadas foram escolhidas para a execução das tarefas implementadas. Essas classes permitiram a escrita e leitura assíncronas entre o computador e o sistema. Foi ainda criada uma classe *Office* com o objectivo de processar as mensagens recebidas do sistema guardar algumas informações nomeadamente o estado de ocupação de cada secretária, os níveis mínimos de iluminação para os estados ocupado e desocupado e os custos de cada nó.
- Processamento: Uma vez que os *IO services* são *thread-safe* é necessária a criação de um único serviço que é usado em duas *threads* distintas. Uma processa a leitura de comandos da porta série outra processa as escritas.

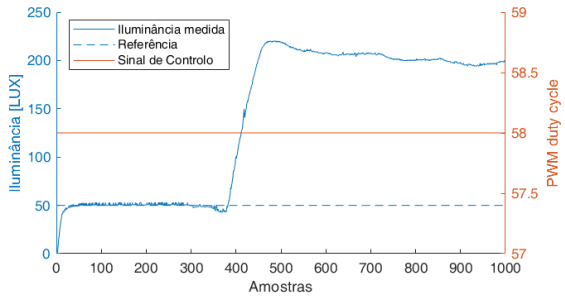
5 Resultados Experimentais e Discussão

5.1 Primeira Fase

Na primeira fase de implementação, foi testado apenas um controlador individual numa luminária a funcionar.

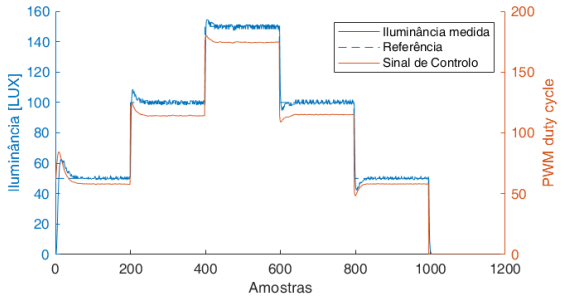


(a) Resposta do controlador feedforward a uma sequência de steps. Experiência realizada durante um período de 12 segundos ($T_s = 10ms$).

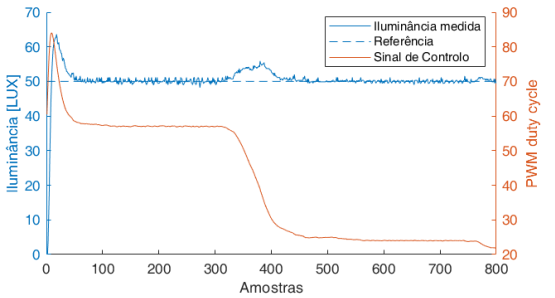


(b) Reação do Controlador FeedForward a perturbações. Experiência realizada durante um período de 8 segundos ($T_s = 10ms$).

Figura 10: Resultados Controlador FeedForward. PWM *duty cycle* representado numa escala de 0 a 255.



(a) Resposta do controlador feedforward e feedback a uma sequência de steps. Experiência realizada durante um período de 12 segundos ($T_s = 10ms$).



(b) Reação do controlador feedforward e feedback a perturbações. Experiência realizada durante um período de 8 segundos ($T_s = 10ms$).

Figura 11: Resultados do controlador FeedBack e FeedForward. PWM *duty cycle* representado numa escala de 0 a 255.

O sistema, tal como explicado na secção 4.5, inicia com um processo de calibração. Uma vez que só existe uma luminária no sistema, a luminária apenas determina o seu ganho K e a sua iluminação externa o, começando de seguida o processo de controlo de iluminância dentro da caixa.

5.1.1 FeedForward

Inicialmente o sistema foi testado utilizando apenas o controlador FeedForward. Como explicado anteriormente este é um controlador rápido mas pouco preciso, uma vez que a sua performance está muito dependente da existência de um bom modelo do sistema. Para testar este controlador foi gerado um sinal de referência que era definido e alterado através da interface do computador com o Arduino. A iluminação medida dentro da caixa foi registada assim como o sinal de controlo. Na figura 10 estão os resultados da analise ao controlador FeedForward. Como é possível ver existe um pequeno erro de seguimento que o controlado é incapaz de corrigir qualquer que seja o valor de referencia pretendido. Esta é uma das características do controlador FeedForward, já abordada na secção 4, que por não usar qualquer mecanismo de retorção é incapaz de corrigir erros de seguimento. Apesar desta incapacidade de eliminar este tipo de erros o erro médio quadrático entre o valor de iluminação medido e a referencia pretendida foi de apenas $28.49LUX$. Este valor não é de estranhar uma vez que o modelo do sistema usado para a construção do controlador de feedforward é apenas um ganho proporcional que é determinado todas as vezes que o sistema é ligado, fazendo com que o erro

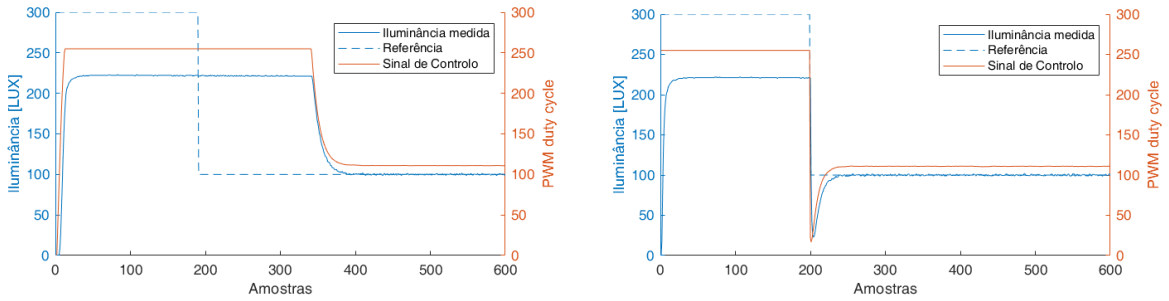
do modelo seja pequeno.
Para testar a robustez do controlador a perturbações externas, foi introduzida iluminação externa na caixa abrindo uma "janela"na caixa. Analisando a figura 10b é possível observar o efeito da introdução da iluminação externa, através da abertura da janela, ocorre um pouco antes da aquisição da amostra 400 ou seja dos 4 segundos. Se o controlador fosse robusto deveria agir aquando da introdução da perturbação externa, diminuir a iluminação para conseguir seguir a referência. No entanto como é possível ver na figura 10b o sinal de controlo mantém-se constante independentemente da existência de perturbações. Mais uma vez este resultado deve-se ao facto de não existir retroacção neste tipo de controlador.
Para concluir, este controlador apesar de ser rápido é inadequado para o problema em questão uma vez que não é robusto a perturbações externas. Para aproveitar a característica positiva deste controlador, a rapidez, e colmatar a negativa, não rejeição de perturbações externas, este controlador é combinado com um controlador de FeedBack.

5.1.2 FeedForward e FeedBack

Como explicado na secção 4, foi implementado um controlador proporcional integral para resolver os problemas que o controlador FeedForward sozinho não consegue resolver.
Os ganhos do controlador, presentes na equação 28, foram determinados empiricamente após várias tentativas em que o comportamento do controlador foi analisado. Após este procedimento chegaram-se aos valores $K_i = 5$ e $K_p = 2$. Para testar este controlador foi gerado um sinal de referência que era definido e alterado através da interface do computador com o arduino. A iluminação medida dentro da caixa foi registada assim como o sinal de controlo.
Analisando a figura 11a é possível ver que a iluminação medida no interior da caixa segue a referencia sem praticamente qualquer erro de seguimento, sendo o erro médio quadrático entre o valor de iluminação medido e a referencia pretendida igual a $25.84LUX$. Este valor é menor que o valor obtido apenas com o controlador FeedForward o que demostra uma melhoria com o uso deste controlador.
É possível ainda verificar, tal como previsto na secção 4, que o controlador apresenta uma sobre-reação a mudanças bruscas na referencia. Estas sobre-reações originam regimes transitórios significativos que tem grande influencia no valor do erro médio quadrático obtido. Como explicado nessa secção este facto está relacionado com o facto de as componentes FeedBack e FeedForward estarem a funcionar de forma desacoplada. Os resultados da solução proposta para a resolução deste problema encontram-se mais adiante na secção 5.1.3.
A reação deste controlador à introdução de iluminação externa na caixa é apresentada na figura 11b. Após a aquisição da amostra 300, ou seja dos 3 segundos, foi aberta uma janela na caixa, levando a um aumento transitório da iluminação medida. No entanto, a iluminação rapidamente retorna ao valor de referência, uma vez que o efeito de retroação do controlador permite compensar esta perturbação reduzindo o sinal de controlo.

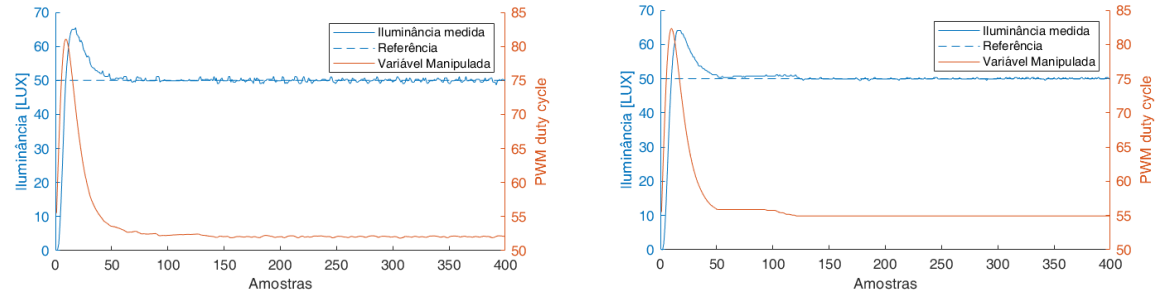
5.1.3 Melhorias no Controlador

Foram realizados testes de modo a demonstrar as melhorias de performance introduzidas pelas seguintes funcionalidades: *Anti-Windup*, *deadzone*, filtro passa-baixo e simulador.
O efeito do *Anti-Windup* é observado na figura 12. A experiência realizada consistiu em inicializar o sistema com um valor de referência de iluminância muito elevado (300 LUX), que se encontra fora dos limites de atuação do LED. Cerca de 2s depois do início da experiência, foi definida uma nova referência de 100 LUX. Quando o *Anti-Windup* não está ativo, o sistema demora bastante tempo a convergir para a nova referência pois está a descarregar o erro acumulado (figura 12a). No entanto, quando é adicionado *Anti-Windup*, a convergência é bastante mais rápida (figure 12b).
Na figura 13, é demonstrada a redução das oscilações no sinal de controlo, resultante da introdução de uma *deadzone* e do filtro passa-baixo descrito na secção 4.3.4. Estas funcionalidades permitem manter o valor do sinal de controlo praticamente constante quando as condições do sistema não sofrem alterações, reduzindo o fenómeno de *flickering*.
Por fim, foi adicionado ao sistema um simulador da resposta ao controlador feedforward. É possível observar na figura 14a uma sobreelevação inicial no sinal de controlo e na iluminância medida, e também quando é mudada a referência. Isto ocorre porque o controlador de feedback atua tentando compensar sozinho o erro inicial, uma vez que não tem conhecimento do efeito do termo de feedfoward no sistema. Ao simular este efeito, o erro na entrada do controlador de feedback é reduzido, levando a uma diminuição das sobreelevações



(a) Sem *Anti-Windup* (b) Com *Anti-Windup*

Figura 12: Melhorias no controlador com adição de Anti Wind-up. Experiência realizada durante um período de 6 segundos ($T_s = 10ms$). PWM *duty cycle* representado numa escala de 0 a 255.



(a) Sem Dead-Zone e Filtro PB (b) Com Dead-Zone e Filtro PB

Figura 13: Melhorias no controlador com adição da *deadzone* e Filtro Passa Baixo. Experiência realizada durante um período de 4 segundos ($T_s = 10ms$). PWM *duty cycle* representado numa escala de 0 a 255.

observadas (figura 14b).

5.1.4 *Jitter* e tempo de processamento

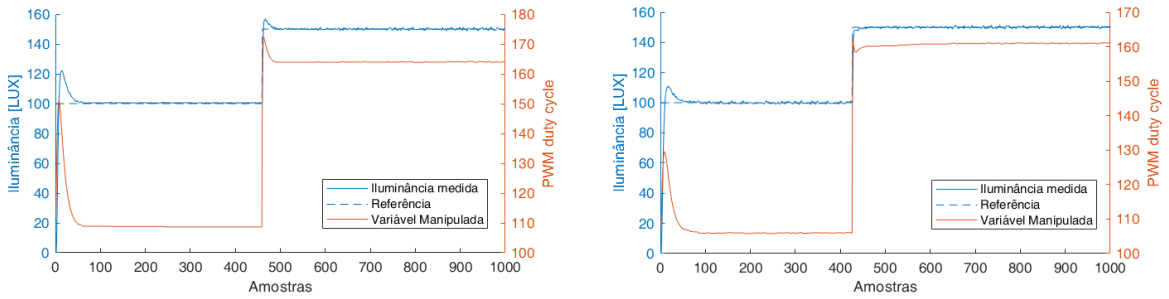
De forma a caraterizar o jitter observado, foi verificando o valor do período de amostragem. Este não se mantém sempre em 10 ms, variando aproximadamente entre 9.2 ms e 10.3 ms. Em cada período de amostragem, as ações do controlador demoram aproximadamente 1.9 ms.

5.2 Segunda Fase

Após a integração das três luminárias no sistema, é necessário testar o sistema de controlo distribuído. Para quantificar a sua performance são utilizadas as métricas descritas na secção 2, nomeadamente, energia consumida, erro de visibilidade e erro de *flicker*.

5.2.1 Controlo Cooperativo *vs* Controlo Não Cooperativo

Em primeiro lugar foi comparada a performance dos sistemas de controlo cooperativo e não cooperativo. Em ambos os casos o sistema foi inicializado considerando todas as secretárias desocupadas, ao fim de 10 segundos foi definida a ocupação da luminária 1 e ao fim de 20 segundos foi definida a ocupação da luminária 2. A resposta observada na figura 15 é semelhante em ambos os casos, uma vez que também no caso não cooperativo a iluminância medida e o sinal de sinal de controlo convergem rapidamente para valores estáveis. No entanto, isto não significa que estes sejam óptimos do ponto de vista do consumo de energia. De facto, a energia acumulada total é ligeiramente inferior quando são usadas as soluções do algoritmo de optimização, como mostra a tabela 1. Apesar de na luminária 1 o consumo de energia com controlo cooperativo ser ligeiramente superior, este é inferior nas outras duas,



(a) Sem Simulador

(b) Com Simulador

Figura 14: Melhorias no controlador com a adição do Simulador. Experiência realizada durante um período de 10 segundos ($T_s = 10ms$). PWM *duty cycle* representado numa escala de 0 a 255.

	Energia acumulada [J]		
	Controlo não Cooperativo	Controlo Cooperativo (c1=c2=c3=1)	Controlo Cooperativo (c1=c3=1; c2=3)
Luminária 1	8.0739	8.2497	10.9835
Luminária 2	6.8414	6.5480	0.9775
Luminária 3	3.5928	3.4765	14.9885
Total	20.5081	18.2742	26.9494

Tabela 1: Resultados do consumo de energia do sistema na experiência da figura 15 (com custos simétricos) e da mesma experiência com custos assimétricos

levando a que globalmente a energia consumida seja inferior. Observando a figura 16, verifica-se que, tal como esperado, o valor de energia está continuamente a aumentar, sendo este aumento tanto mais rápido quando maior o valor do sinal de controlo, uma vez que este corresponde ao valor de PWM aplicado no LED.
Por outro lado, verifica-se um desempenho globalmente melhor em termos de Erro de *flicker* no caso não cooperativo. Isto deve-se a dois fatores: em primeiro lugar, quando é alterada uma referência a solução do algoritmo não consegue ser calculada antes do início do próximo ciclo de controlo. Além disso, as soluções do algoritmo podem não ser aplicadas simultaneamente nas três luminárias. Assim, ocorre um curto período de pequenas oscilações após a aplicação das soluções do algoritmo, levando a um aumento do *flicker*. O erro de visibilidade tem um comportamento semelhante nas três luminárias, tendendo para um valor próximo de zero, como ilustra a figura 17. Verifica-se ainda que nas luminárias 1 e 3 o erro de visibilidade é em geral inferior quando é usado controlo cooperativo, demonstrando outro benefício deste tipo de controlo.

5.2.2 Controlo Cooperativo com custos assimétricos

Foi também realizada a mesma experiência considerando que a fonte de iluminação de uma das luminárias gasta mais energia que as restantes. Em particular, definiu-se o custo do nó correspondente à luminária 2 três vezes superior ao das luminárias 1 e 3. Os resultados relativos ao consumo de energia nesta experiência são apresentados na tabela 1. Verifica-se que o consumo de energia na luminára 2 é bastante inferior às outras duas (apenas 0.9775 J). Isto ocorre porque o custo da energia fornecida por esta luminária é superior ao das restantes. Assim, o algoritmo determina soluções óptimas que consistem num valor de *duty cycle* perto de zero para a luminária 2 e valores superiores aos representados na figura 15 para as restantes. O aumento de energia fornecida pelas restantes luminárias garante assim que seja alcançado o valor de referência de iluminância na luminária 2. O *tradeoff* é um valor de iluminância acima da referência nas luminárias 1 e 3.

5.2.3 Tempo de processamento

Observou-se experimentalmente que o tempo de necessário para obter a solução óptima do algoritmo ao fim de 50 iterações é cerca de 1.1s e que o cálculo dos resultados de uma iteração do *Consensus* num dado nó necessita de 0.586ms, o que reforça a importância da máquina de estados implementada.

6 Conclusão

O controlador distribuído proposto foi implementado com sucesso, incluindo não só os controladores PI locais mas também o algoritmo de optimização *Consensus*, que permite ao sistema realizar um Controlo Cooperativo da iluminação nas várias luminárias.
Verificou-se que todas as funcionalidades introduzidas nos controladores locais foram de extrema importância para obter uma boa performance do mesmo. Nomeadamente, foi possível garantir um rápido seguimento da referência de iluminação bem como robustez face a perturbações externas com os controladores feedback e feedfoward e a deadzone e o filtro passa-baixo permitiram praticamente anular oscilações no sinal de controlo. O anti-windup e o simulador foram também fundamentais para eliminar períodos transitórios em que a iluminação não correspondia à desejada.
No sistema de controlo cooperativo, a comunicação de soluções do algoritmo entre nós é um ponto essencial, e que levantou vários problemas ao longo do projeto. Nomeadamente, pelo facto de o sistema funcionar de forma assíncrona, foi necessário adoptar estratégias para que as iterações avancem em todos os nós de modo a permitir que o algoritmo converja para a solução óptima. Após diversas melhorias, a implementação aqui apresentada permite que esta seja obtida num tempo razoável e aplicada pelo controlador de modo minimizar a energia consumida. Os testes realizados ao sistema permitiram obter resultados promissores, em particular a diminuição da energia gasta pelo sistema de controlo cooperativo face ao sistema não cooperativo. Prevê-se assim que aplicando este protótipo a uma situação real se poderia minimizar o consumo de energia num ambiente de escritório.
Por fim, o desenvolvimento de uma aplicação para interface com o sistema mostrou-se essencial para a realização de testes ao sistema de modo a demonstrar o seu funcionamento.

6.1 Contribuições para o Relatório

Apesar de todos os membros do grupo terem colaborado nas diversas secções, a divisão de tarefas foi a seguinte:

- Maria Brites - Arquitetura do Sistema, Controlador Distribuído, Comunicações Resultados Experimentais e Discussão, Conclusão
- João Lopes - Introdução, Definição do Problema, Modelo da Luminária, Controlador Local, Inicialização e Calibração, Resultados Experimentais e Discussão
- Gabriel Antero - Resumo, Solução Proposta, Controlador Distribuído, Aplicação

Referências

[1] Arduino mcp2515 can interface library. `https://github.com/autowp/arduino-mcp2515`.

[2] European Standard EN 12464-1. *Lighting of indoor workplaces*. April 2013.

[3] Alexandre Bernardino. The consensus algorithm applied to the distributed illumination control problem. *Real-Time Distributed Control Systems 2019/20*, Module 22, 2019.

[4] David Caicedo and Ashish Pandharipande. Distributed illumination control with local sensing and actuation in networked lighting systems. 2013.

[5] LIDA OPTICAL & ELECTRONIC CO., LTD. *CdS photoconductive cells*.

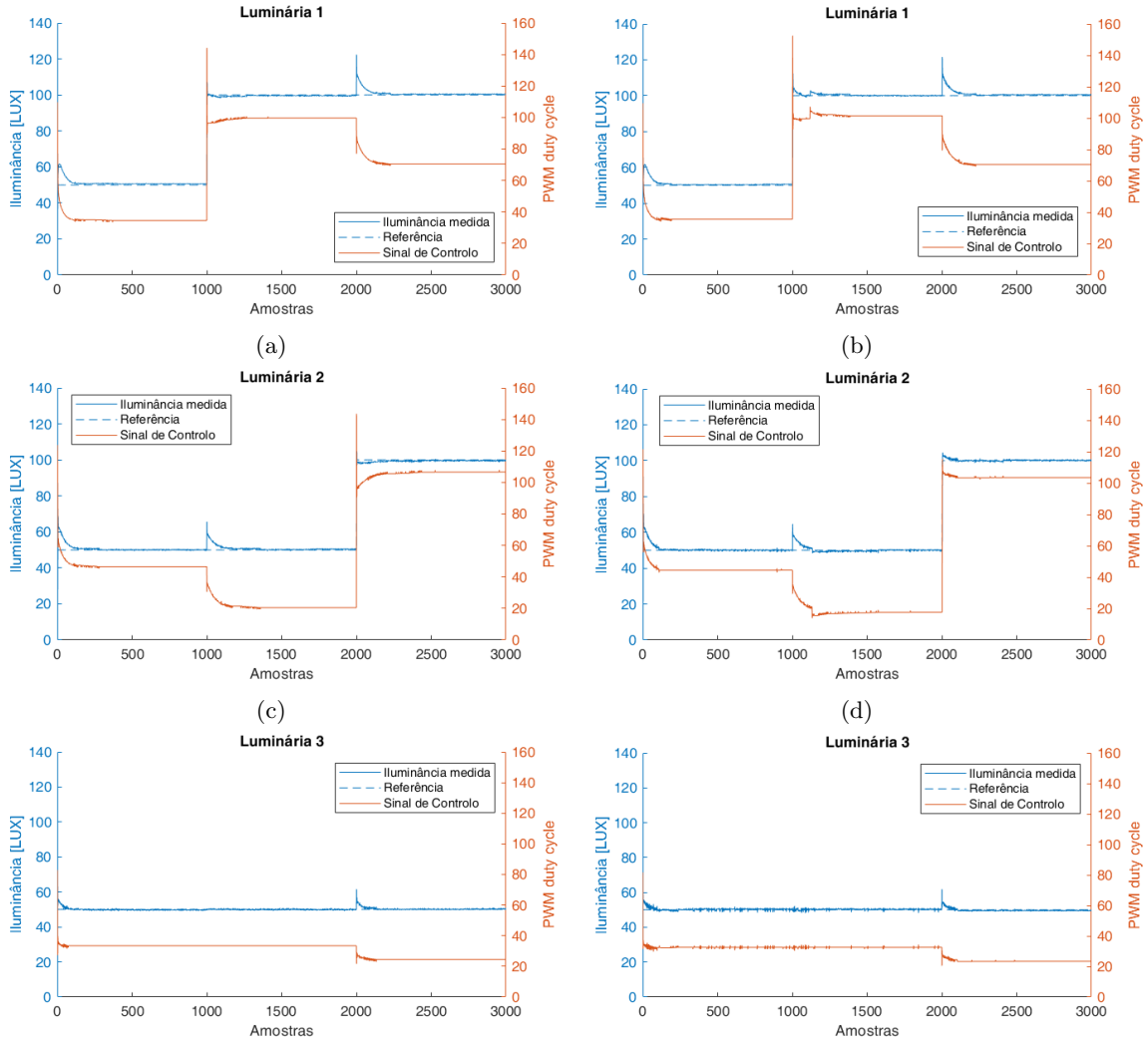
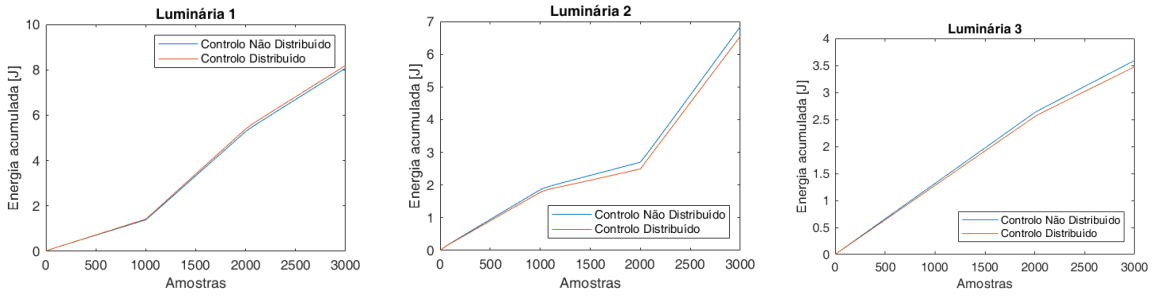
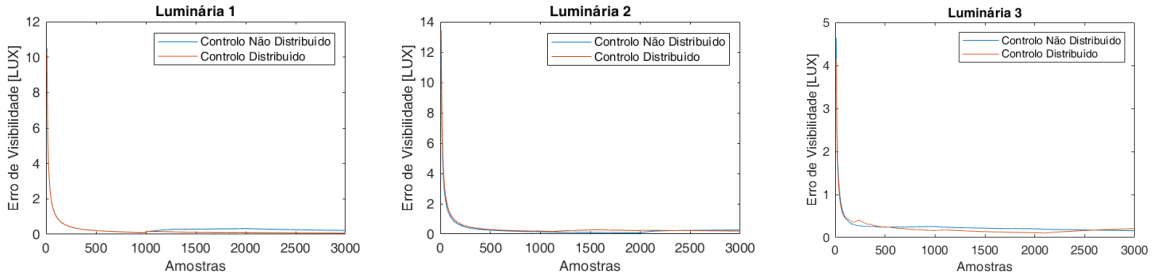


Figura 15: Comparação entre controlo distribuído e não distribuído; (a), (c), (e): resultados obtidos usando apenas os controladores individuais; (b), (d), (f): resultados obtidos com o algoritmo *Consensus*, em adição aos controladores individuais. Experiência realizada durante um período de 30 segundos ($T_s = 10ms$). PWM *duty cycle* representado numa escala de 0 a 255.



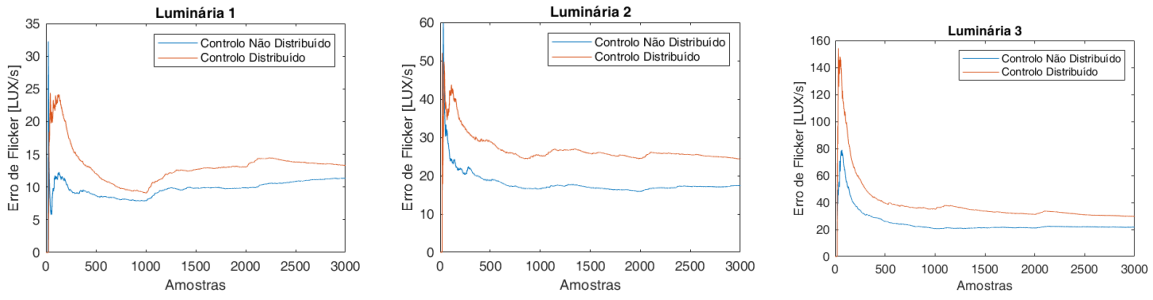
(a) (b) (c)

Figura 16: Energia acumulada nas três luminárias, durante a experiência da figura 15



(a) (b) (c)

Figura 17: Erro de visibilidade nas três luminárias, durante a experiência da figura 15



(a) (b) (c)

Figura 18: Erro de *flicker* nas três luminárias, durante a experiência da figura 15